

Sprawozdania
Wojciech Wnuk
IO 7.15

Zaawansowane programowanie w Javie
VII semestr
Rok akademicki: 2023/24

LABORATORIUM 1. STRUMIENIE I PLIKI

Zadanie 1.1. Operacje na strumieniach

```
Main.java x
1 package com.company;
2
3 import java.io.*;
4 import java.util.*;
5 import java.util.stream.*;
6
7 public class Main {
8     public static void main(String[] args) {
9
10         // Zadanie 1.1: Operacje na strumieniach
11
12         List<String> przedmioty = new ArrayList<>();
13         przedmioty.addAll(Arrays.asList("Integracja systemow", "Interakcja czlowiek-komputer",
14             "Programowanie aplikacji mobilnych na platforme Android",
15             "Programowanie aplikacji mobilnych na platforme iOS"));
16
17         Stream<String> strumienPrzedmiotow = przedmioty.stream();
18         List<Integer> listaOczen = strumienPrzedmiotow
19             .filter(przedmiot -> !przedmiot.contains("Zaaw"))
20             .map(przedmiot -> {
21                 return (int) (Math.random() * 4) + 2;
22             })
23             .collect(Collectors.toList());
24
25         listaOczen.forEach(ocena -> System.out.println("1.1: Ocena: " + ocena));
26
27         Map<Integer, Long> liczbaPowtorzen = listaOczen.stream()
28             .collect(Collectors.groupingBy(Integer::intValue, Collectors.counting()));
29
30         liczbaPowtorzen.forEach((ocena, liczba) -> {
31             if (liczba > 1) {
32                 System.out.println("Ocena " + ocena + " powtarza się " + liczba + " razy.");
33             }
34         });
35     }
```

Zadanie 1.2. Operacje na plikach

```
5
6 // Zadanie 1.2: Operacje na plikach
7
8 List<String> ocenyIZapis = listaOcen
9     .stream()
10     .map(ocena -> "Ocena: " + ocena)
11     .collect(Collectors.toList());
12
13 Collections.sort(ocenyIZapis);
14
15 try {
16     BufferedWriter writer = new BufferedWriter(new FileWriter(fileName: "oceny.txt"));
17     for (String ocena : ocenyIZapis) {
18         writer.write(ocena);
19         writer.newLine();
20     }
21     writer.close();
22 } catch (IOException e) {
23     e.printStackTrace();
24 }
25
```

Zadanie 1.3. Operacje na plikach

```
// Zadanie 1.3: Operacje na plikach

List<String> ocenyIZPliku = new ArrayList<>();

try {
    BufferedReader reader = new BufferedReader(new FileReader("oceny.txt"));
    String linia;
    while ((linia = reader.readLine()) != null) {
        ocenyIZPliku.add(linia);
    }
    reader.close();
} catch (IOException e) {
    e.printStackTrace();
}

double srednia = ocenyIZPliku.stream()
    .mapToDouble(ocena -> Integer.parseInt(ocena.substring(7)))
    .average()
    .orElse(0);

Optional<String> najlepszaOcena = ocenyIZPliku.stream()
    .max(Comparator.comparingInt(ocena -> Integer.parseInt(ocena.substring(7))));
Optional<String> najgorszaOcena = ocenyIZPliku.stream()
    .min(Comparator.comparingInt(ocena -> Integer.parseInt(ocena.substring(7))));

System.out.println("Średnia ocen: " + srednia);
System.out.println("Najlepsza ocena: " + najlepszaOcena.orElse("Brak ocen"));
System.out.println("Najgorsza ocena: " + najgorszaOcena.orElse("Brak ocen"));
}
```

Wyniki działania kodu:

```
"C:\Program Files\Java\jdk-17\bin
1.1: Ocena: 4
1.1: Ocena: 3
1.1: Ocena: 5
1.1: Ocena: 4
Ocena 4 powtarza się 2 razy.
Średnia ocen: 4.0
Najlepsza ocena: Ocena: 5
Najgorsza ocena: Ocena: 3
```

```
oceny — Notatnik
Plik  Edycja  Format  Widok  F
Ocena: 3
Ocena: 4
Ocena: 4
Ocena: 5
```

Wnioski:

Wykorzystanie strumieni w Javie pozwala na wygodne i efektywne przetwarzanie danych, zarówno w przypadku operacji na kolekcjach, jak i operacji na plikach. Funkcje takie jak filter, map i collect są użytecznymi narzędziami do manipulacji danymi w strumieniach, umożliwiając filtrowanie, transformację i zbieranie wyników w odpowiednich strukturach danych.

LABORATORIUM 2. WYKORZYSTANIE PLIKÓW XML.